FAILURE ENVIRONMENT ANALYSIS TOOL APPLICATIONS

Ginger L. Pack
NASA Johnson Space Center
Houston, Texas 77058

David B. Wadsworth
Lockheed Engineering and Sciences Company
Houston, Texas 77058

ABSTRACT

Understanding risks and avoiding failure are daily concerns for the women and men of NASA. Although NASA's mission propels us to push the limits of technology, and though the risks are considerable, the NASA community has instilled within it, the determination to preserve the integrity of the systems upon which our mission and, our employees lives and well-being depend. One of the ways this is being done is by expanding and improving the tools used to perform risk assessment. The Failure Environment Analysis Tool (FEAT) was developed to help engineers and analysts more thoroughly and reliably conduct risk assessment and failure analysis. FEAT accomplishes this by providing answers to questions regarding what might have caused a particular failure; or, conversely, what effect the occurrence of a failure might have on an entire system. Additionally, FEAT can determine what common causes could have resulted in other combinations of failures. FEAT will even help determine the vulnerability of a system to failures, in light of reduced capability. FEAT also is useful in training personnel who must develop an understanding of particular systems. FEAT facilitates training on system behavior, by providing an automated environment in which to conduct "what-if" evaluation. These types of analyses make FEAT a valuable tool for engineers and operations personnel in the design, analysis, and operation of NASA space systems.

INTRODUCTION

FEAT was developed as part of an effort to find ways to better identify and understand potential failures that threaten the integrity of NASA systems. Past and current methods of failure assessment consists of developing often enormous amounts of documentation in the form of Failure Mode Effect Analysis (FMEA) worksheets. Engineers create these worksheets by attempting to exhaustively enumerate potential system failures and consequences. Hazards analysis is performed in a similar manner; experts are gathered together and are asked to brainstorm about the hazardous manifestations of various failures. System knowledge and experience are necessary for ensuring the comprehensiveness of this approach. However there are troubling drawbacks to this technique. First, there exists the difficulty of anticipating every scenario. Analysis is also inherently constrained by the limits of actual experience. Further, such methods lack consistency and do not enforce a standard level of coverage. Although there is certainly much to be credited to knowledge acquired through experience, it is not sufficient to avoid unanticipated interactions which may lead unexpectedly to undesirable consequences. As many industries have learned, sometimes experience comes at too high a cost. Those at NASA have been looking for better ways to anticipate failure and for tools to assist in "designing out" potential problems. FEAT was developed to address this problem.

TECHNICAL APPROACH

FEAT is a software application that uses directed graphs or, digraphs, to analyze failure paths and failure event propagation. The behavior of the systems to be analyzed is represented as a digraph. Then, the digraph model of the system, is used by FEAT to answer questions concerning the cause and effects of events which are captured in the model. Therefore, the first step in using FEAT is to create the digraph model of the system in which one is interested. Once FEAT has analyzed the digraph, it has the information it needs to perform cause and effect analysis.

What are digraphs? Directed graphs are graphs that consists of a set of vertices and a set of edges, where there is an edge from one vertex *a* to another vertex *b*. The vertices are drawn as circles and the edges are drawn as arrows. The direction of the arrows indicates a causal relationship between the vertices (see figure 1). The vertex

from which the edge begins, is called its source; and the vertex at which the edge terminates, is called its target. Direct graph theory is an accepted and established area of mathematical study. Therefore we will only introduce it in this paper, to the extent necessary for an understanding of how it is used in FEAT. The interested reader may find further information by consulting the literature.

Figure 1

The structure of the digraph can be represented by a matrix, and consequently can be easily implemented in a computer. The conversion from digraph to matrix is straightforward and is illustrated below in figure 2. This matrix is called the *adjacency* matrix (reference 1), and is the basis from which other information about the graph can be derived. The matrix of the graph is obtained by entering either zero or one, depending on whether or not an edge connects two vertices. The presence of an edge from a to b in figure 1, indicates an entry of one (1) into the corresponding matrix entry. However, since there is no edge from a to c, a zero (0) would be entered in the corresponding matrix entry.

	a	b	c
a	0	1	0
b	0	0	1
c	0	0	0

Figure 2

Additional information can be added to the digraph, by applying logical operators to express conditional statements. FEAT uses AND and OR operators to accomplish its analysis. The AND operator is represented on the graph as a vertical bar with a horizontally placed arrow at its center. An OR operator is simply two or more edges whose target is the same vertex. Theses operators [figure 3], and their use in FEAT [figures 4 & 5], are described below.

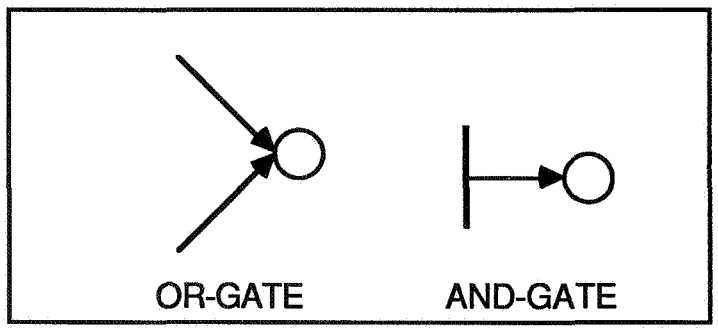

Figure 3

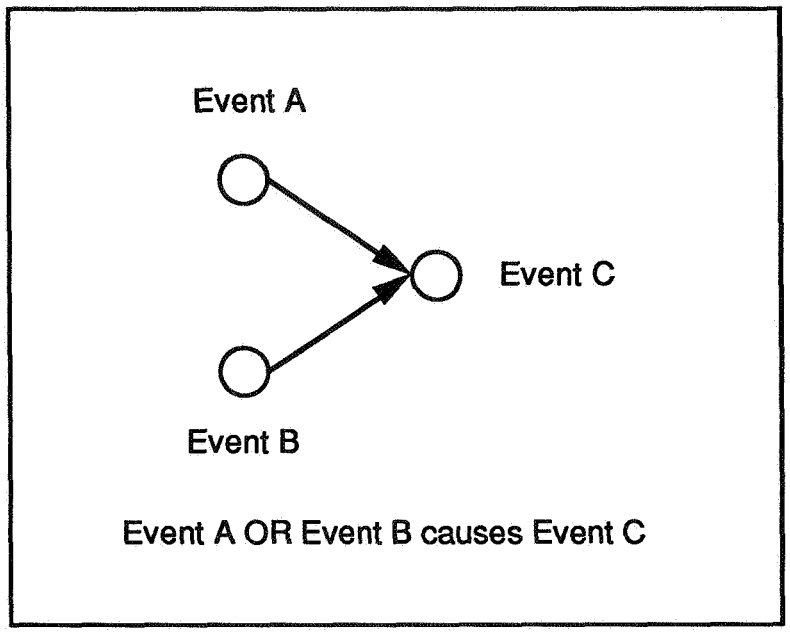

Figure 4

The "AND" gate is shown in Figure 5. The AND gate is used when both event A and Event B must occur in order for Event C to occur. Conversely, if only Event A occurs or, if only Event B occurs, then Event C does not occur.

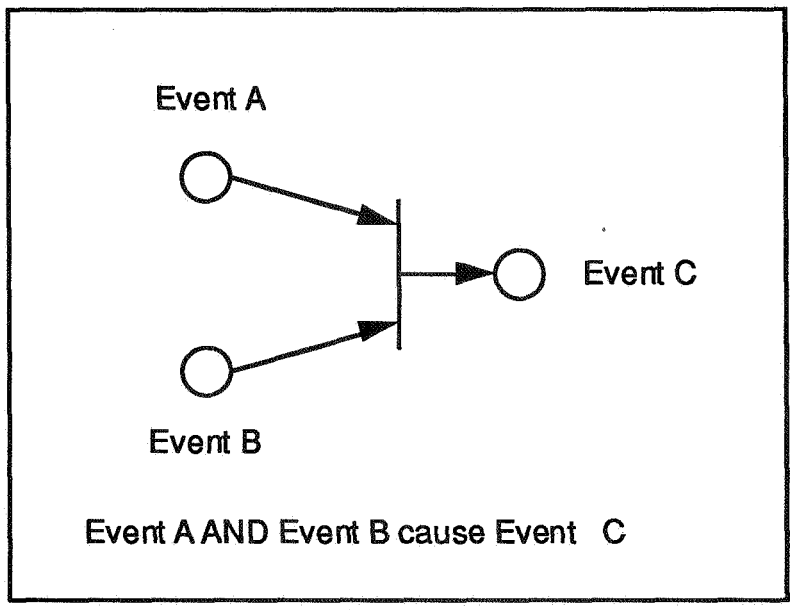

Figure 5

Analytical Capabilities The *reachability* of an event refers to whether there is a path by which other events in the digraph can be reached. A given event is said to reach another, if the first event can cause the second through some path of the graph. Using the adjacency information derived from the digraph, reachability can be computed for every event and pair of events in the digraph. Analysis can be conducted upstream or downstream from an event node. (References 2, 3 and 4 provide a much more detailed discussion of digraphs and reachability.)

Reachability information allows FEAT to answer the following questions about a modeled system:

A. What happens to the system if "Event A (and Event B and Event C and ...)" occurs?
B. What are the possible causes of "Event A"?
C. What common cause could account for the simultaneous indication of numerous events?
D. What is the susceptibility of the system to new events given that one or more events has already occurred, or the system has been reconfigured due, for example, to maintenance?

Digraph Example The following example demonstrates how a digraph might be implemented for a light and switch. The digraph provides a methodical way in which to express the topology and behavior of a system. It is worth noting that the digraph itself may have various constructions for the same information contained in it, depending on who created it. Different modelers may lay out the digraph differently. However, for a properly constructed digraph, the same information will be captured. In the following example [figures 6 & 7], power source A provides current to switch A which connects to the bulb. Similarly, power source B can energize the bulb.

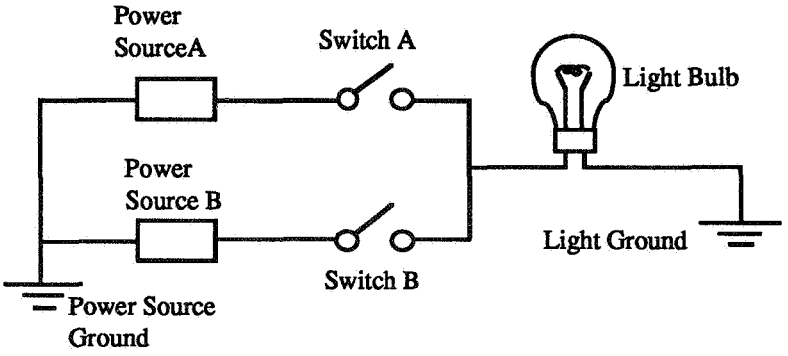

Figure 6
Light bulb and Power Source Schematic

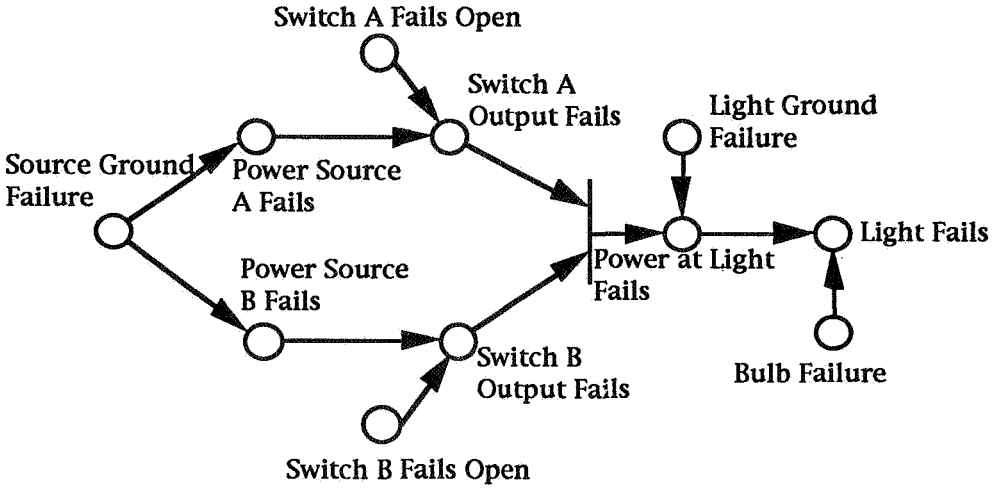

Figure 7
Digraph of Light bulb Schematic

- If "Power Source A Fails" or "Switch A Fails Open" then "Switch A Output Fails". This is an example of OR logic and is shown in the digraph by the arrows leading into "Switch A Output Fails".

- If output from both switches A and B fail, then they will cause the "Power at Light to Fail". This logic appears as an AND gate on the digraph (the vertical line). In this case, the AND gate reflects redundancy designed into a system.

Why digraphs?

Directed graphs are useful because they visually depict the logical topology and dependency relationships of physical and conceptual systems and processes. Because they capture causal effects between events, they can be used to describe system behavior. Directed graphs are also easily converted into a matrix and, because of this, can be readily analyzed in a computer. Creating and laying out the digraph of a system, also formalizes the method of evaluation during the analytical process, and provides a standard representation convention. Finally, digraph analysis is mathematically sound, since methods for determining connectivity paths of the digraph vertices can be mathematically proved.

DIRECTED GRAPHS AND FEAT

Digraph construction is facilitated by use of an editor specifically designed for the task. Such an editor is included in the FEAT package which consists of two programs: Digraph Editor and FEAT.

Digraph Editor

The Digraph Editor facilitates construction of the digraph model by allowing the user to create event nodes, edges, and the logic operators, and to connect and arrange them into a digraph. Event nodes and edges are laid out and connected using the logic operators. The pieces that make up a digraph are supplied in a digraph toolbox from which items may be selected. These items are placed on the screen and arranged to produce the system digraph.

Other information is needed to complete the digraph and to make it usable by FEAT. Event nodes have an associated text block, which includes information that will identify the event node to FEAT, describe the event for the user, and relate the event to a drawing which contains the component to which the event pertains. This information is extracted from tables that the user creates. Digraph Editor uses the tables to automatically generate a mnemonic reference that FEAT will use to identify the event.

Digraph Editor also provides a number of tools for validating and verifying the model as it is being developed. Digraph Editor will check tables for duplicate entries, check nodes for incorrect form, and determine whether a selected node has a duplicate in the digraph. Digraph Editor also contains an algorithm that allows the user to analyze small or incomplete digraphs, while still in the editor. Once the digraph is completed and the paths in it are analyzed, FEAT can return answers to questions regarding the behavior of the modeled system.

Currently, digraph models are created manually by selecting and arranging digraph components; the modeler must interpret drawings and other sources of information to generate the digraphs. This is a laborious task. Consequently, efforts are underway to develop methods to automatically translate schematics and drawings into corresponding digraph models.

Digraph Editor is currently only available for the Macintosh II class of computer.

FEAT

FEAT is the portion of the package that analyzes single or multiple digraphs, and graphically displays causes and effects of events. Propagation results are shown both on the digraphs and on an another associated graphical representation, such as a schematic or block diagram. FEAT uses a multi-step algorithm, described in Reference 2, to compute reachability for each event and pair of events in the digraphs. Events are identified to FEAT through the mnemonic that is generated by Digraph Editor. Queries about the behavior of the system are made by selecting events and telling FEAT to return all of the causes of that event (targeting), or by telling FEAT to return all of the effects of that event (sourcing). FEAT displays all of the single events, and all pairs of events that may cause a selected event. Multiple events may also be selected and analyzed. FEAT allows some events to be temporarily removed from the analysis so that answers can be obtained about a reconfigured system.

FEAT also contains a feature which allows users to attach to a schematic, formatted database information and graphics. In this way, component descriptions, parts lists, drawings, etc, may be displayed in conjunction with a schematic.

One of the major advantages of FEAT, as discussed in Reference 2, is that it allows the analysis of very large systems. Large systems can be digraphed by creating and connecting a series of smaller digraphs. FEAT understands when propagation occurs across the digraphs.

Planned enhancements to FEAT include the following: increasing the speed with which reachability is computed by improving FEAT's computational algorithm; provision of a method for computing and displaying probabilities of events occurring; and computation and display of the time it takes for an event to propagate through the graph.

FEAT is currently available for the Macintosh II class of computer and for UNIX/X-Windows/OSF-Motif systems. No programming skill is required to use FEAT. However, a course in digraph modeling is quite helpful in learning how to construct system models.

DIGRAPHS AT NASA

Why NASA chose digraphs

NASA's interest in digraphs began as part of the Shuttle Integrated Risk Assessment Project (SIRA). SIRA was initiated in the wake of the Challenger accident, in an effort to find better ways of assessing risk and preventing failure. Digraphs support such analysis by providing end to end cause and effect analysis of modeled systems. Digraphs also provide a standard and methodical approach for conducting safety analysis and risk assessment. Digraphs capture information in an easily retrievable format, and facilitate the transfer of design information. FEAT takes advantage of these characteristics in a way that aids engineers and analysts with design, assists safety engineers with risk assessment, and promotes understanding of system behavior, thereby making FEAT a good tool for training inexperienced persons.

What has been done at NASA?

The first system to which digraph analysis was applied was the Space Shuttle Main Engine System (SSME). Since then, acceptance of digraphs and the use of FEAT has extended in several directions. Most recently, FEAT has been formally released to the Space Station Freedom Program (SSFP) Technical Management Information System (TMIS), as Digraph Data System (DDS) Release 1.0. DDS will, through TMIS, be available to SSF Engineering and Integration, SSF Combined Control Center, and the various Work Packages and their contractors. A Macintosh Powerbook version of FEAT will be deployed as a Development Test Objective (DTO) on the STS-52 flight scheduled for October 1992. Reliability and Maintainability personnel at NASA-JSC, are using FEAT to construct a model of the Simplified Aid for Extra-Vehicular Activity (EVA) Rescue (SAFER). FEAT is also being used to model the redesigned Servo Power Amplifier (SPA) for the Remote Manipulator System (RMS).

Proponents have used FEAT for a variety of analytical tasks, such as Fault Tolerance Analysis and Redundancy Management (FT/RM), Fault Detection, Isolation, and Recovery (FDIR), and "What-If" analysis. Within the Space Station Freedom Program, FEAT is being used in the performance of Integrated Risk Assessment for the station, which includes Failure Mode and Effects Analysis (FMEA), Hazards Analysis (HA), and FT/RM. FEAT has also been established as a baselined tool in the Mission Operations Combined Control Center, where flight controllers will use FEAT models to assist with real-time monitoring tasks. FEAT's role is expanding in both Space Station and in Space Shuttle.

Space Station The Space Station Engineering Integration Contractor (SSEIC), is using FEAT to perform integrated risk assessment. This tasks consists of performing the analysis to assure the station design is safe, reliable, and has an acceptable level of risk (reference 5). The space station design consists of modules designed and built by the United States, and of modules which will be designed and built by NASA's international partners. The work to be performed by NASA is divided into four Work Packages distributed among different centers. Additionally, a variety of contractors are working in support of the Work Packages. Consequently, system integration is a paramount concern of the program. SSEIC is tasked with ensuring the integration of these various factions and is using digraph-based FEAT, to work the integration problem. Specifically, FEAT supports the

following areas of the Integrated Risk Assessment process:

1. Reliability Analysis
2. Safety Analysis
3. Integrated Risk Analysis
4. Integrated Risk Assessment

The models being developed for the station Integrated Risk Assessment will eventually be provided to Mission Operations personnel for use in FDIR of the on-orbit station.

Space Shuttle FEAT is scheduled to fly on STS-52 as a Detailed Test Objective (DTO). A FEAT model of the S-band Communications System has been installed on an Apple™ Powerbook™, which will be flown aboard the shuttle. Astronauts will use the model to perform on-board fault isolation for the S-band Communication System. They will be able to configure the model to match the actual S-band system configuration, and then will use FEAT to identify possible causes of failures of the S-band system.

FUTURE APPLICATIONS OF DIGRAPHS

Digraphs are gaining acceptance, within the NASA community, as a viable method for conducting many kinds of analysis. Space Station Freedom Program and Operations, has mandated the use of digraph analysis for the Space Station Level II Integration effort; and many others are beginning to take up the banner. Some of the potential areas of application include the following:

Fault Isolation/Testability

FEAT's ability to model and analyze system failures make it a natural candidate for fault isolation efforts. If a failure event occurs, FEAT can display all of the possible single and paired causes for that event. However, in a large system, potential causes can be enormous in number. A method of pruning the list of possible causes is then necessary. Sensor information associated with the system can be used to remove candidate causes which occur downstream of a known nominal condition. Incorporation of sensor data, into the analysis, can help to reduce the number of candidate failures to a manageable sum. Then using traditional techniques, further isolation can be accomplished. Figure 8 shows an example of such a case.

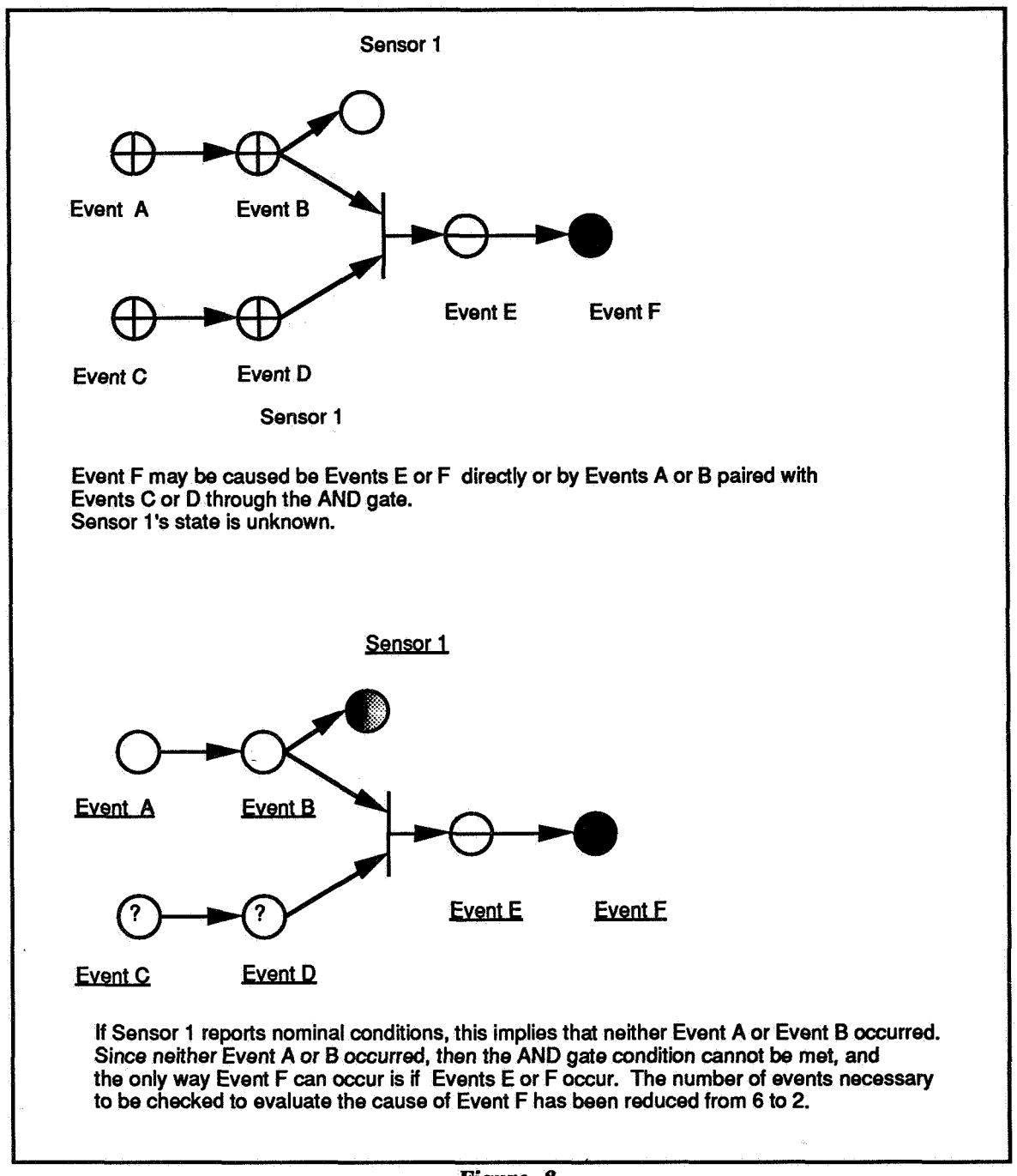

Figure 8

Sensor data may also be combined with FEAT to identify the potential for cascading alarms. For instance, if a fault occurs downstream from a sensor, the sensors upstream will eventually alarm as a result of the fault. FEAT can show the effects of a fault on the downstream sensors.

This solution is being implemented by NASA, in an extension of FEAT, called Extended Real-time FEAT (ERF). ERF automatically prunes the list of possible faults, according to sensor information. ERF is being developed as a part of the FDIR system for the On-orbit Control Center Complex. Mission Controllers will use ERF to resolve off-nominal system behavior, by reducing the potential number of failure causes.

FEAT developers are pursing the possibility of incorporating, or interfacing with, a testability analysis tool which will help to evaluate sensor coverage in systems, and make recommendations regarding appropriate sensor locations. ERF is dependent upon adequate sensor information and proper placement of the sensors. Properly placed sensors provide information to quickly and accurately locate faults. The combination of FEAT, ERF and testability tools will make a very powerful fault isolation system.

Temporal Analysis

Not every event immediately affects the next downstream event. There may be appreciable delays within an event and between events. For example, an inappropriately shut valve, may not for some time, cause the pressure in the system to rise to an unacceptable level. In such a situation, time delay is an important aspect of calculating the potential failure space.

This issue will be addressed in FEAT when a modification is made to Digraph Editor to allow modelers to include time delays within events, and delays between events. FEAT will then compute the maximum and minimum time delay between selected events. This capability will be supplied in a future version of FEAT.

Software Modeling

Physical systems are not the only candidates for digraph analysis. Software functions and data flow can be modeled as well. Particularly, the flow and effect of invalid/improper data can be modeled. This can provide insight to the designer in determining mission critical software functions. Additionally, the effect of invalid data on other system functions (both software and hardware) may be shown. For instance, a software functional component that generates invalid data as an event; may then provide that data to other software and hardware as an invalid data input event. FEAT can be used to model these behaviors too.

Design Evaluation and Redundancy Management

Digraph models can be used to determine whether or not a system design provides sufficient redundancy. Maintenance and configuration effects on the system, can be evaluated by selectively removing (setting) components from the system. The reconfigured system can then be evaluated for induced single and paired events. This can be particularly useful in determining new vulnerabilities after a system has encountered failures and/or has portions of the system secured for maintenance.

FEAT contributes to design evaluation by rapidly displaying all single events caused by the event of interest, and all pairs of events that will result in that event. Unexpected single point common cause events are also quickly identified. As the design is modified to provide additional redundancy, the digraph model can be updated to reflect the changes, and the new set of single events and pairs of events can be evaluated.

Logistics Analysis

Logistics analysis addresses corrective and preventive maintenance tasks, and determines the kinds and numbers of repair parts needed for a system. This type of analysis is associated with the reliability and availability (reference 6), of systems. Reliability is defined as the measure of the mean time between failure (MTBF) and, concerns the probability that a system will operate over a specified period of time. No provision is made for repair when calculating reliability. Availability varies from reliability, in that it is a measure of the mean time to repair (MTTR), or, the probability that the system will operate over a period of time considering that something can be done to restore functionality lost as a result of a failure. How system repairs can be supported, or supportability, is important to determining availability. If repairs can be made instantaneously, availability is increased. However, long delays between failure and repairs makes the system proportionally less available.

FEAT models can help to identify critical components and the effect of their failure upon the system. Digraph models of the system can, along with specific part reliability, help to determine priorities for inventory stocks, and schedules for maintenance. Spare parts inventories are a major factor in determining supportability. For example, spares for parts that cause single point common cause events should have higher priority for stocking than parts that contribute to pairs of events.

Maintainability concerns the time it takes to remove and replace a component. Digraph models can identify components prone to low reliability, and single common cause failure. Designers can then either improve the reliability of the component or ensure that such items are accessible and easily replaced.

SUMMARY

As NASA continues to search for better and innovative approaches to new and old problems, directed graph analysis has emerged as a viable addition to the methods applied to Risk Assessment. Directed graphs are a well established area of mathematical study and analysis, and provide an easily comprehendible visual representation of cause and effect relationships. Conversion of the digraph to an equivalent matrix is straightforward, and allows analysis of digraphs to be mathematically calculated and verified. The nature of matrices also makes them ideally suited for computerized calculations, which in turn provides a vehicle for automating the task of risk assessment and failure analysis.

FEAT uses directed graph theory to provide engineers and analysts with a powerful and flexible automated analytic helper. FEAT can provide end to end analysis of cause and effect events. Very large systems can be modeled in modules, then connected to form the entire system. This feature also allows digraphs to be arranged in mix and match fashion. FEAT can detect and return information about single point failure vulnerability, failure event pairs, common cause events, and reduced capability analysis. FEAT shows the results of event propagation on system schematics and on the associated digraph. Digraph Editor provides a helpful way for the analyst to create digraphs.

The FEAT Project is funded by the NASA Space Station Freedom (SSF) Advanced Programs Development Office (Code MT) and the SSF Program Office (Code MS).

REFERENCES

1. L. Levy, *Discrete Structures of Computer Science.* John Wiley & Sons, 1980.
2. R. Stevenson, J. Miller and M. Austin. Failure Environment Analysis Tool (FEAT) Development Status. *AIAA Computing in Aerospace VIII, AIAA-91-3803.* October 1991.
3. I. Sacks. Digraph Matrix Analysis. *IEEE Transactions on Reliability, Vol. R-34, No. 5.* December 1985.
4. I. Sacks, G. Keller, and R. Rauch. Application of Digraph Matrix Analysis to the Space Station. *RDA , Logicon, R & D Associates, RDS-TR-148400-001.* September 1987.
5. J. Schier. Integrated Risk Assessment (IRA): Defining the Level II Safety & Reliability Job and Implementation Plan using Digraphs. *Unpublished Grumman presentation.* September, 1992.
6. B. Blanchard. *Logistics Engineering and Management.* 4th Edition. Prentis-Hall, Inc. Englewood Cliff, NJ. 1992.

BIBLIOGRAPHY

D. Haasl, N. Roberts, W. Vesely, F. Goldberg. *Fault Tree Handbook.* GPO Sales Program, U.S. Nuclear Regulatory Commission, Washington, DC. 1981.

J.Pearl, *Probabilistic Reasoning in Intelligent Systems.* Morgan and Kauffman, 1988.